Exploring Content Popularity in Information-Centric Networks

Andriana Ioannou, Stefan Weber

School of Computer Science and Statistics Trinity College, Dublin 2, Ireland

Abstract: Information-Centric Networking (ICN), an alternative architecture to the current Internet infrastructure, focuses on the distribution and retrieval of content by employing caches in a network to reduce network traffic. The employment of caches may be accomplished using graph-based and content-based criteria such as the position of a node in a network and content popularity. The contribution of this paper lies on the characterization of content popularity for on-path in-network caching. To this end, four dynamic approaches for identifying content popularity are evaluated via simulations. Content popularity may be determined per chunk or per object, calculated by the number of requests for a content against the sum of requests or the maximum number of requests. Based on the results, chunk-based approaches provide 23% more accurate content popularity calculations than object-based approaches. In addition, approaches that are based on the comparison of a content against the maximum number of requests have been shown to be more accurate than the alternatives.

Keywords: network distributed architectures; future internet; information-centric networks; caching technologies; on-path caching; content popularity

I. INTRODUCTION

Information-Centric Networking (ICN), an alternative to the host-centric model of the current Internet architecture, focuses on content distribution and retrieval. In ICN, content sources publish content to a content notification service, i.e. a name resolution or a name-based routing service, while clients subscribe to a content notification service to retrieve content [1]. In contrast to existing content-centric technologies such as Content Delivery Networks (CDNs) and Peer-to-Peer (P2P) networks, ICN is not application-specific or proprietary.

ICN architectures decouple content from specific hosts using *content identifiers*. Content identifiers may correspond to content resources such as web pages, files or parts of a content resource, chunks or packets and should involve no information that would bind the content to a specific location [2], [3]. If this constraint is met, content can be freely replicated.

In contrast to the traditional host-centric model, ICN introduces the capacity to cache content within the infrastructure; routers are equipped with cache memory and enabled with a caching capability. As a result, multiple copies of a content may be available at various locations within a network. Approaches to caching can be categorized into *off-path caching* and *on-path caching* with regard to the location of caches [4], [5].

Off-path caching aims to replicate content within a network regardless of the forwarding path. Off-path caching is usually centralized and involves a great amount of information collected and advertised into a content notification service. The off-path caching problem is equivalent to the CDN content replication and web cache placement problems [6], [5].

On-path caching is integrated to the forwarding procedure which in turn impedes a number of challenges, e.g. caching decisions can only be applied on content that is propagated on the delivery path(s), on nodes that are lying on the delivery path(s), caching mechanisms are bounded by the on-line speed requirements of the delivery process where the overhead of monitoring, collection of statistical information or advertisement of the cached content to a content notification service may not be acceptable or feasible, caching topologies are arbitrary.

On-path caching approaches may be categorized depending on the criterion used for the caching decision, i.e. random probability [7], [8], topological information such as the distance of a node from a source [9], [10], [11] and content popularity [12], [13], [14]. Even though, content popularity has been defined as a key feature to the performance of a caching algorithm and the network [15], [14], [16], no consideration has been given to the way content popularity should be calculated.

In this paper, we focus on the identification of content popularity for on-path caching. We propose four dynamic approaches for calculating content popularity on per chunk basis and per object basis which we evaluate via simulations. Content popularity may be calculated by the number of requests for a content against either the total number of requests or the maximum number of requests for a content. Our goal is to examine the accuracy and limitations of each approach.

The remainder of this paper is structured as follows. Section II presents the existing work on on-path caching and content popularity as well as the justification of using dynamic approaches. Section III describes the dynamic algorithms being proposed. Section IV presents the simulation model used for evaluation. Section V presents the evaluation results of the algorithms against one another and against the content popularity experienced by the producer as well as indications for future work. Section VII is dedicated to the conclusions.

This article presents that the employment of caches may be accomplished using content-based criteria - content popularity. To achieve this goal, the authors propose four dynamic algorithms that calculate content popularity on per chunk basis and per object basis.

II. Related Work

Existing on-path caching approaches can be divided into *random probabilistic approaches*, *graph-based approaches* and *content-based approaches*, depending on the caching criteria used. These approaches aim to distribute copies of content along the delivery paths in order to reduce delivery latencies while limiting the

Table I *Taxonomy of the existing on-path caching algorithms.*

Proposed Technique	Comparison Technique	Caching Category	Description
BC [9]	CE^2, UniCache	graph-based	graph-centrality metric
BC, CC, DC, EC, GC, SC [17]	BC, CC, DC, EC, GC, SC	graph-based	graph-centrality metric
CAC [12]	BC, CE^2	content-based & graph-based	content popularity & link throughput
CE^2 [8]	-	random probabilistic	random probability
FIX(p) [7]	CE^2	random probabilistic	random probability
LCD [14]	CE^2, FIX(p)	content-based & graph-based	content popularity & node position
LeafNode [10]	CE^2	graph-based	node position
PCP [11]	CE^2	content-based & graph-based	content popularity & node position
ProbCache [18]	CE^2, FIX(p), LCD	graph-based	hop count
UniCache [13], [9]	BC, WAVE	random probabilistic	random probability
WAVE [13]	CE^2, FIX(0.1), UniCache	content-based & graph-based	content popularity & node position

number of replicas. A brief taxonomy of the approaches can be found in Table I.

Random probabilistic approaches base the probability to create a replica at a node on a fixed value. A fixed value, *FIX(p)*, may correspond either to an arbitrary number [7], [8] or to the number of nodes in a delivery path e.g. *UniCache* [13], [9]. These simplistic approaches neglect the variable nature of content requests and network topologies.

Graph-based approaches such as *Betweeness Centrality (BC)*, *Degree Centrality (DC)* [9], [17] and *LeafNode* [10] aim to react to the topology of the network by taking into account the nodes on a delivery path. BC takes into account the number of times a node is included in the shortest paths between sources and consumers, DC is based on the number of connections of a node and LeafNode caches a copy of the requested content at the last node of the delivery path. These approaches may lead to advantages in terms of server hits or hop count rates when requests are well distributed, but they neglect the frequency and distribution of content requests.

Measurement studies in web caching have highlighted the problem of *cache pollution* due to *one-timer* objects [19]. One-timer objects are objects requested only once while cache pollution is the case where one-timer objects are cached, decreasing the performance of a network; cache pollution results in an increase of the cache miss rates and the network traffic rates. According to the same studies, one-timer objects correspond to 45-75% of the requests. As on-path caching is expected to serve a higher number of contents than object replication mechanisms, under more severe restrictions such as cache memory availability [7], [20], cache pollution prevention becomes an important prerequisite that random probabilistic and graph-based on-path caching approaches fail to address.

In order to address this, recently proposed approaches on on-path caching have been based on content popularity [12], [13], [14]; according to a number of existing works in ICN caching [15], [21], [22], [14], [16], content popularity has been proven to be a crucial factor to the performance of a caching algorithm and the performance of a network. The idea of popularity-based caching is that popular contents will satisfy a higher number of requests. Therefore, caching popular contents should be preferred while caching unpopular contents should be prevented. Depending on the criterion to which content popularity is calculated, content popularity may be divided into *static content-popularity* and *dynamic content-popularity*.

Static content-popularity approaches require the definition of a threshold h. Contents with a number of requests higher than h are considered to be popular while contents with a number of requests lower than h are considered to be unpopular [21], [22], [16], [11]. Unpopular contents are excluded from the caching decision. Due to the volatile nature of ICN architectures, we expect the definition of threshold h to be challenging; static content-popularity approaches, usually, result in out of date calculations and unutilized cache capacity [21].

Dynamic content-popularity is defined as the number of requests for a content during a time interval Δt [12], [23], [24]. A common technique is to define an infinite Δt [12], [13], [14]. Dynamic-content popularity may be applied on per path basis or on per node basis, i.e. *per-path dynamic content-popularity* and *per-node dynamic content-popularity*, respectively.

As an example, *Leave Copy Down (LCD)* [14] and *WAVE* [13] algorithms are applied on a delivery path and do not implicitly compare the popularity of a content against the rest of the contents. LCD caches a copy of the requested content one hop closer to a consumer each time a content request arrives while WAVE caches a linearly increased number of chunks each time an object request arrives, i.e. upon the n^{th} request for an object on node n, 2^{n-1} chunks will be cached at node *n-1*, 2^{n-2} chunks will be cached at node *n-2*, etc. *CAC* [12] is applied on per node basis and implicitly compares the popularity of a content to the popularity of the rest of the contents, by comparing the number of requests for a content to the sum of requests for the rest of the contents.

The difference between per-path content-popularity and per- node content popularity is that in the former, caching is applied on at least one node on the delivery path upon the arrival of the first content request while in the latter no such guarantee exists; caching depends on the content popularity observed on a node. As a result, per-path content-popularity approaches fail to address the cache pollution problem. In the next section, we propose four dynamic content-popularity algorithms calculated per node basis and discuss their efficiency.

III. Dynamic Content Popularity Algorithms

To explain the dynamic content-popularity algorithms further, a set of notations are first defined. Let n denote the node performing the caching decision and i denote the piece of content of object o on which the caching decision is applied, i.e. $i \epsilon o$. Let $R_{n,i}$ denote the number of requests recorded on node n for piece of content i and $R_{n,o}$ denote the number of requests for object o with $\sum_{i'=1}^{I} R_{n,i'}$ being the total number of requests. Let Δt_i denote the time interval for which the aforementioned metrics are calculated.

To explain the use of time interval Δt_i, a short description of the ICN architecture is provided. In ICN [8], nodes are capable of keeping track of the requested contents via a *Pending Interest Table (PIT)* table. A PIT table keeps a list of the incoming and outgoing interfaces with regard to content requests, called *Interests*. A PIT entry is created upon the first arrival of an Interest, with the following Interests being suppressed to the same entry. This procedure is called *suppression of requests*. Upon the arrival of content, i.e. a Data packet, the forwarding system uses the interface list held by the entry to deliver the content. The entry is then deleted. In order not to introduce any overhead to the infrastructure of a node, such as in the case of an infinite time interval [12] which is based on the creation of infinite entries, we define Δt_i as the time for which an entry with regard to content i is alive. We then define the dynamic content-popularity algorithms as follows:

$$Popularity_{(n,i)} = \frac{\frac{R_{n,i}}{\Delta_{t_i}}}{\sum_{\forall i' \epsilon I} \frac{R_{n,i'}}{\Delta_{t_i}}} \tag{1}$$

$$Popularity_{(n,i)} = \frac{\frac{R_{n,i}}{\Delta_{t_i}}}{max_{\forall i' \epsilon I} \frac{R_{n,i'}}{\Delta_{t_i}}} \tag{2}$$

$$Popularity_{(n,i)} = \frac{\frac{R_{n,o}}{\Delta_{t_i}}}{\sum_{\forall i' \epsilon I} \frac{R_{n,i'}}{\Delta_{t_i}}} \tag{3}$$

$$Popularity_{(n,i)} = \frac{\frac{R_{n,o}}{\Delta_{t_i}}}{avg_{\forall i' \epsilon I} \frac{R_{n,i'}}{\Delta_{t_i}}} \tag{4}$$

Equations (1-4) correspond to an implicit comparison of a content against the remaining contents, concluding to a value that lies in the range [0, 1]. Equations (1-4) correspond to the abbreviations: *Chunk-Sum (CS)*, *Chunk-Maximum (CM)*, *Object-Sum (OS)* and *Object-Average (OA)*, respectively.

The Chunk-Sum algorithm (1), calculates the popularity of a piece of content i based on the number of requests for content i during time interval Δt_i, divided by the total number of requests during the same time interval on node n. However, since a content competes against the sum of contents requested during the same time interval and given the high catalog sizes that ICN architectures are expected to serve, only contents with very high request rates will be considered to be popular, while the majority of contents will be discarded. As an example, if the number of requests for content i during time interval Δt_i is equal to 10 and the total number of requests during the same time interval is equal to 1000, the popularity of content i will be equal to 0.01, which is substantially low. Hence, alternative approaches to the way content popularity is calculated should be taken into account.

Towards this direction, the Chunk-Maximum algorithm (2), calculates the popularity of a piece of content *i* based on the number of requests for content *i* during time interval Δt_i, divided by the maximum number of requests for any piece of content observed during the same time interval on node *n*. The advantage of this approach lies to the fact that contents are competing against each other to determine their popularity, hence, a better approximation of the contents popularity is provided. Following the previous example and assuming that the maximum number of requests for a content is equal to 50, the popularity of content i will be equal to 0.2. The disadvantage of this approach lies to its complexity. To accurately implement a Chunk-Maximum algorithm, the inclusion and update of an extra field at each PIT entry holding the maximum request rates observed during time interval Δt_i is required. However, this complexity may not be manageable in terms of calculation resources of a router. As an alternative, the approximation of the maximum request rates for a content using a defined a priori Δt, adjustable to the network demands may be considered.

Since ICN architectures are able to operate either on a chunk or object naming granularity, the determination of whether content popularity should be defined per chunk or per object is still an open question. To solve this confusion, two more algorithms regarding the calculation of content popularity are considered where the statistic metrics are calculated per object. The Object-Sum algorithm (3), corresponds to the formula used by CAC algorithm for the calculation of content popularity [12], as explained in section I. The Object-Sum algorithm calculates the popularity of a piece of content *i* based on the number of requests for object *o* during time interval Δt_i divided by the total number of requests during the same time interval on node *n*. This way, each piece of content follows the content popularity of the object where it belongs, which in turn increases the number of requests of a content against the remaining contents. However, Object-Sum algorithms may still suffer from the same drawback defined for Chunk-Sum algorithms. Thus, an alternative against the most popular object seen during the same time interval should be acknowledged.

However, since the implementation of an Object-Max algorithm would introduce substantial complexity and overhead into not only a router but a simulation study as well, an Object- Average algorithm (4) is alternatively considered. To justify our choice, we briefly state the complexity ovehead that an Object-Max algorithm would introduce. An implementation of an Object-Max algorithm would require the update of each PIT entry with regard to the calculation of the maximum number of requests per object during the time interval Δt_i. In order to achieve this, a copy of each PIT entry for each Δt_i time interval that an object entry is valid should be maintained and updated upon either the arrival of an Interest or the deletion of a PIT entry. To avoid this computational overhead and complexity, the Object-Average algorithm (4), calculates the popularity of a piece of content *i* based on the number of requests for object *o* during time interval Δt_i divided by the average number of requests of an object during the same time interval on node *n*. An evaluation of the aforementioned approaches is provided in the following section.

IV. System Model

The evaluation is based on the ndnSIM simulator [25], an ns-3 module that adopts the Named Data Networking (NDN) communication model [8]. A summary of the parameters used can be found in Table II. A real network topology, *Exodus AS-3967* is used [26]. The network topology, Fig.1, consists of 94 nodes, i.e. 39 backbone nodes (blue) and 58 gateway nodes (green). Each node is equipped with a NDN stack. Consumer and producer applications can only be installed on a gateway node. In particular, one producer, chosen based on the metric of degree-centrality is assumed; a node with degree-centrality 5 is chosen, where 1 is the minimum and 14 is the maximum. The evaluation is based on traffic characteristics extracted from YouTube traces, with the catalog size being 10^8 [27] and the content store (CS) size *csi* being 10GB [7]. However, due to memory restrictions [14], both parameters are reduced by a magnitude of 10^4, i.e. to 10^4 and 1MB, respectively. Object sizes follow a normal distribution of mean 10MB and standard deviation 9.8MB [28]. To study the effect of popularity on the performance of the algorithms, a Zipf-Mandelbrot (Z-M) object popularity distribution of $\alpha\epsilon\{0.8, 1.0, 1.5, 2.0, 2.5\}$ and $q\epsilon\{0.5, 5\}$ [27], [29] is defined. Contents in CS are replaced using a Least Recently Used (LRU) policy [8].

Finallizing the system model, a mean value of 200 consumers is installed on each gateway, following a uniform distribution. A consumer generates object requests. Each object request corresponds to a sequence of chunk requests, equal to the size of the object divided by the chunk size, 10KB [28]. Request arrivals follow an exponential distribution of $\lambda = 1.0$.

V. EVALUATION

In this section, a report of the evaluation results of the algorithms proposed in section III is provided. Due to space limitations and in order to ease readability, only the mean values of the evaluation results are displayed and only

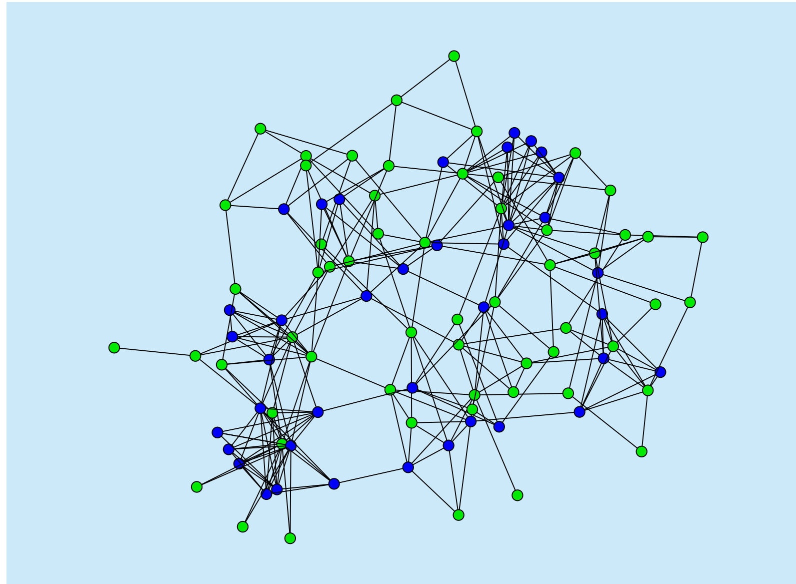

Fig.1 *Exodus topology (AS-3967)*

the abbreviations of the algorithms' names are used, i.e. *CS,CM,OS,OA*. Each evaluation result corresponds to the mean value of 10 simulations with regard to parameters α and *q*. To ease comparison between the algorithms, the average value derived from all evaluation results with respect to each parameter is used.

In order to evaluate the algorithms, a comparison of the popularity of a content determined by a popularity algorithm against the popularity of a content experienced by the producer is performed. To this end, the average popularity of a content recorded on

Table II *Parameters of the system model used for evaluation.*

Parameter	Symbol	Value	Definition
No. of nodes	$\|N\|$	97	Total No. of nodes
No. of backbones	$\|B\|$	39	Total No. of backbone nodes
No. of gateways	$\|G\|$	58	Total No. of gateway nodes
Capacity of links	BW	40GB	Available bandwidth
Catalog Size	$\|O\|$	10000	Total No. of objects
Object Size	o_i	$\forall o_i, i\epsilon\|O\| \sim N(10000KB, 9800KB)$	Size of object *oi* in KB
Chunk Size	Ch	$10KB$	Chunk size in KB
Contents Size	$\|C\|$	$\|C\|\ \sum_{i=1}^{\|O\|} o_i/Ch$	Total No. of chunks
Cache Size	cs_i	$\forall cs_i, i\epsilon\|N\|, cs_i\epsilon\{100\}$	Cache capacity of node *i* in chunks
Consumers Size	u_i	$\forall u_i, i\epsilon\|G\| \sim U(100, 300)$	No. of users on gateway *i*
Rank Parameter	q	$q\epsilon\{0.5, 5\}$	Rank parameter of the Z-M distribution
Zipf Exponent	α	$\alpha\epsilon\{0.8, 1.0, 1.5, 2.0, 2.5\}$	Exponent of the Z-M distribution
Arrival rate	λ	1.0	Exponential request arrival rate
Control window	W	1	No. of requests able to sent with no reply
Simulation time	Time	$100s$	Simulation time used for the evaluation

all nodes in the network is extracted. The popularity experienced by the producer is determined based on the total number of requests for a content during the simulation time. All popularity results are decreasingly sorted and increasingly ranked. A content with the highest popularity value will correspond to a ranking value of 1. The accuracy of a content's popularity is then determined by the difference of its ranking value concluded by a popularity algorithm against its ranking value concluded by the producer. In order to explore whether the accuracy of content popularity calculations is affected by the suppression of requests, a scenario where the suppression of requests is not feasible, i.e. *scenario 1* and a scenario where the suppression of requests operates as described in section III, i.e. *scenario 2*, are evaluated.

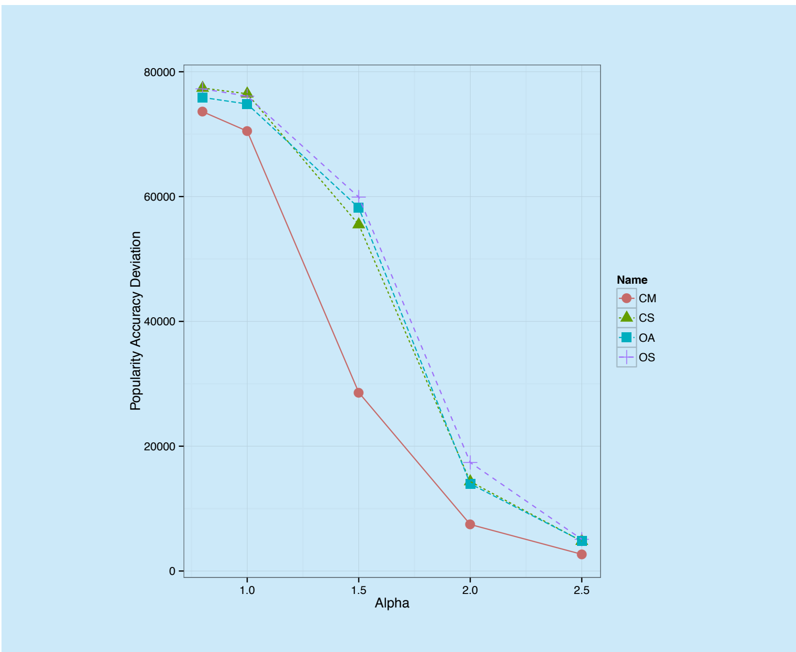

Fig.2 *Popularity accuracy for scenario 1 and q=0.5*

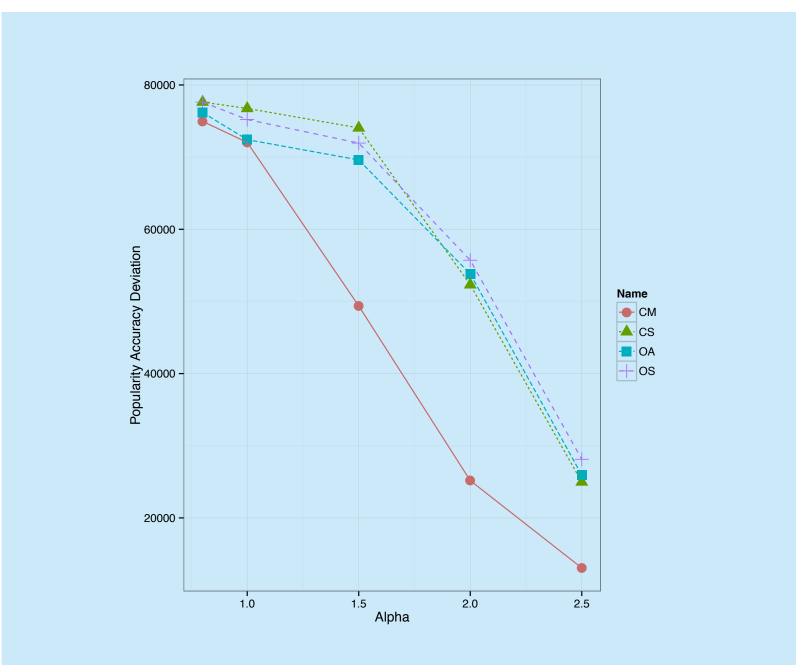

Fig.3 *Popularity accuracy for scenario 1 and q=5*

Fig.2 illustrates the average ranking difference of the popularity algorithms and the producer for parameters q = 0.5 and $\alpha\epsilon\{0.8, 1.0, 1.5, 2.0, 2.5\}$ for scenario 1. According to Fig.2, the accuracy of content popularity determined by CM is considerably higher than the alternatives, i.e. approximately 18% to the average ranking difference while the rest of the algorithms correspond to a minimal difference between 1%-2%. An important point that needs to be highlighted is the lack of OS algorithm to provide comparable performance with regard to the alternatives. As previously stated, OS and CS algorithms may conclude to lower precision than their alternatives due to the comparison of a content against the rest of the contents.

Fig.3 plots the average ranking difference of each approach for parameters q=5 and $\alpha r\{0.8, 1.0, 1.5, 2.0, 2.5\}$ for the same evaluation scenario. According to the plot, CM algorithm exhibits an even lower deviation of 23% of the popularity experienced by the producer compared to the rest of the algorithms. The result suggests that the alteration of parameter q has no significant effect on the behavior of the algorithms.

Fig.4 and Fig.5 correspond to the average content ranking concluded by the popularity algorithms against the producer for the same sets of parameters, for scenario 2. Based on the results, CM and OS exhibit similar results to those displayed in Fig.2 and Fig.3. In contrast, OA and CS algorithms indicate a more distinguishable pattern of their accuracy against the producer and against one another.

More precisely, OA indicates 0.6% and 3% more accurate estimation of content popularity compared to CS, for $q = 0.5$ and $q = 5$, respectively.

Based on the results, one can conclude that the algorithms that base the determination of content popularity on the sum of the remaining contents are outperformed by their alternatives, i.e. CS is outperformed by CM and OS is outperformed by OA. However, whether CS provides a better estimation of content popularity than OA depends on the suppression of content requests. Based on the comparison of Fig.3 and Fig.5, OA provides a more accurate estimation of content popularity than CS when requests suppression is applied. As the suppression of requests is one of the features of ICN architectures, we can safely conclude that OA would correspond to a better choice than CS. Further research is necessary to conclude to the complexity overhead of each approach with regard to the computational resources and time consumption, that will help us to define the advantages, disadvantages and feasibility of each approach. We attempt to fulfill this as a future work.

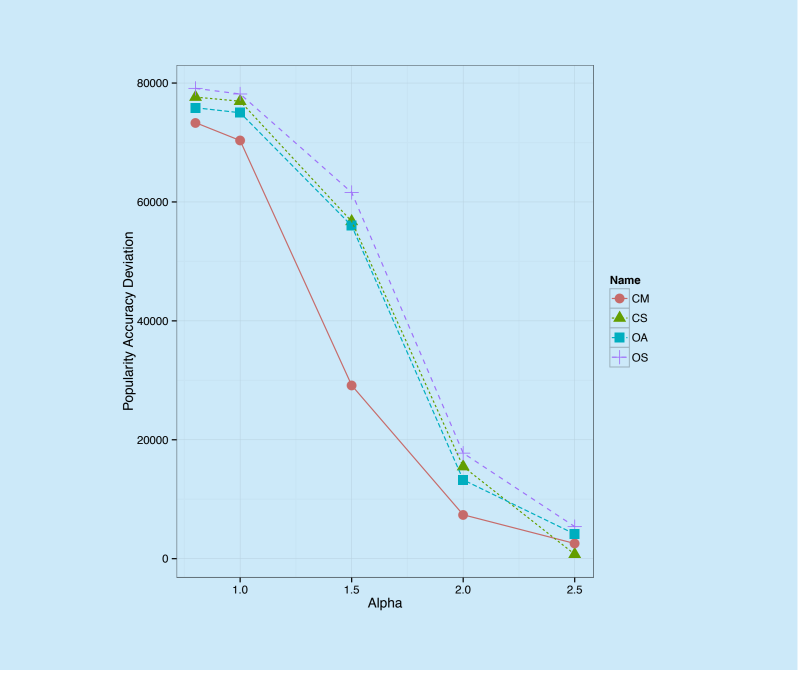

Fig.4 *Popularity accuracy for scenario 2 and q=0.5*

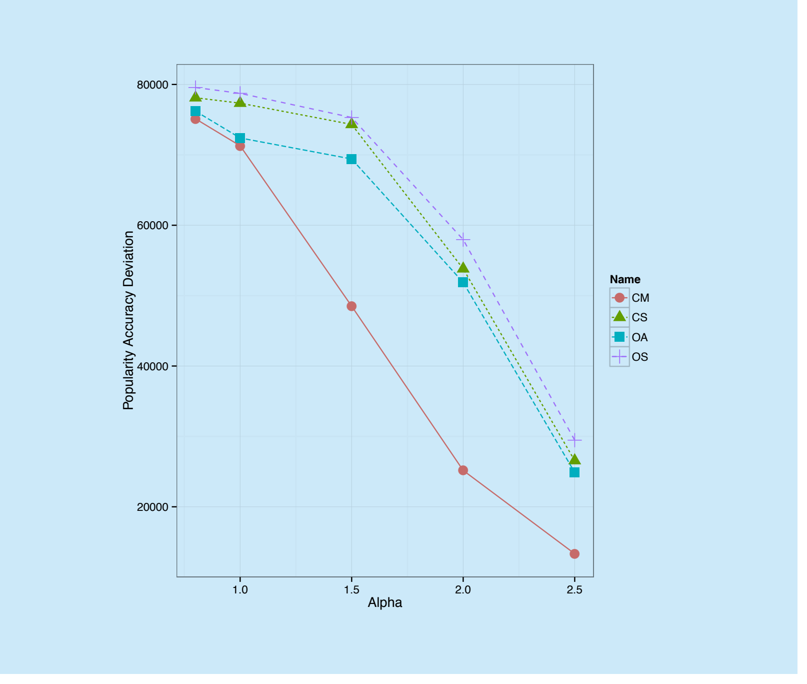

Fig.5 *Popularity accuracy for scenario 2 and q=5*

VI. Conclusion

In this paper, we have described the existing approaches for identifying content popularity in ICN architectures and categorized them against their properties. We have further proposed four dynamic algorithms that calculate content popularity on per chunk basis and per object basis which we evaluated against each other and against the popularity experienced by the producer via simulations. The results indicate that dynamic approaches that are based on the comparison of a content against the maximum or the average content popularity recorded, provide a more accurate estimation with regard to the popularity experienced by the producer, compared to the approaches based on the sum of the contents, while content popularity calculations based on chunks conclude to a better approximation than the ones based on objects. In our future work, we intend to explore the algorithms with regard to computational resources.

Acknowledgments

This work is funded by the Higher Education

Authority (HEA) and co-funded under the European Regional Development Fund (ERDF).

References

[1] P.T. Eugster, P.A. Felber, R. Guerraoui, and A.M. Kermarrec. The many faces of publish/subscribe. *ACM Computing Surveys*, 35(2):114–131, 2003.

[2] H. Balakrishnan, K. Lakshminarayanan, S. Ratnasamy, S. Shenker, I. Stoica, and M. Walfish. A layered naming architecture for the internet. In *Proceedings of the ACM Conference on Applications, Technologies, Architectures and Protocols for Computer Communications (SIGCOMM '04), Portland, OR, USA, August 2004*, pages 343–352.

[3] M. Walfisha, H. Balakrishnana, and S. Shenkerb. Untangling the web from dns. In *Proceedings of the 1st Symposium on Networked Systems Design and Implementation (NSDI '04), San Francisco, CA, USA, March 2004*.

[4] Bengt Ahlgren, Christian Dannewitz, Claudio Imbrenda, Dirk Kutscher, and Bo¨rje Ohlman. A survey of information-centric networking. *IEEE Communications Magazine*, 50(7):26–36, 2012.

[5] George Xylomenos, C Ververidis, V Siris, Nikos Fotiou, Christos Tsilopoulos, Xenofon Vasilakos, K Katsaros, and G Polyzos. A survey of information-centric networking research. *IEEE Communications Surveys and Tutorials*, 16(2):1–26, 2013.

[6] V. Sourlas, P. Flegkas, G.S. Paschos, D. Katsaros, and L. Tassiulas. Storage planning and replica assignment in content-centric publish/subscribe networks. *Elsevier Journal on Computer Networks*, 55(18):4021–4032, 2011.

[7] S. Arianfar, P. Nikander, and J. Ott. On content-centric router design and implications. In *Proceedings of the Re-Architecting the Internet Workshop of the ACM CoNEXT Conference (ReArch '10), Philadelphia, USA, November 2010*.

[8] V. Jacobson, D.K. Smetters, J.D. Thornton, M.F. Plass, N.H. Briggs, and R.L. Braynard. Networking named content. In *Proceedings of the 5th International Conference on Emerging Networking Experiments and Technologies (CoNEXT '09), Rome, Italy, December 2009*, pages 1–12.

[9] W. Chai, D. He, I. Psaras, and G. Pavlou. Cache "less for more" in information-centric networks. In *Proceedings of the 11th International IFIP TC 6 Conference on Networking,Brooklyn, NY, USA, May 2012*, pages 27–40.

[10] V. Sourlas, G.S. Paschos, P. Flegkas, and L. Tassiulas. Caching in content- based publish/subscribe systems. In *Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM '09), Honolulu, HI, USA, November 2009*, pages 1–6.

[11] Jason Min Wang and Brahim Bensaou. Progressive caching in ccn. In *Global Communications Conference (GLOBECOM), 2012 IEEE*, pages 2727–2732. IEEE, 2012.

[12] Mikhail Badov, Anand Seetharam, Jim Kurose, Victor Firoiu, and Soumendra Nanda. Congestion-aware caching and search in information-centric networks. In *Proceedings of the 1st ACM Conference on Information-Centric Networking (ICN'14), Paris, France, September 2014*, pages 37–46.

[13] K. Cho, M. Lee, K. Park, T.T. Kwon, Y. Choi, and S. Pack. Wave: Popularity-based and collaborative in-network caching for content- oriented networks. In *Proceedings of the 1st Workshop on Emerging Design Choices in Name-Oriented Networking (NOMEN '12), Orlando, FL, USA, March 2012*, pages 316–321.

[14] Dario Rossi and Giuseppe Rossini. Caching performance of content centric networks under multi-path routing (and more). Technical report, Telecom ParisTech Ecole, November 2011.

[15] G. Carofiglio, M. Gallo, L. Muscariello, and D. Perino. Modeling data transfer in content-centric networking. In *Proceedings of the 23rd International Teletraffic Congress (ITC'11), San Francisco, CA, USA, September 2011*, pages 111–118.

[16] Edmund Yeh, Tracey Ho, Michael Burd, Ying Cui, and Derek Leong. Vip: A framework for joint dynamic forwarding and caching in named data networks. In *Proceedings of the 1st ACM Conference on Information- Centric Networking (ICN'14), Paris, France, September 2014*, pages 117– 126.

[17] Giuseppe Rossini and Dario Rossi. On sizing ccn content stores by exploiting topological information. In *Proceedings of the 1st Workshop on Emerging Design Choices in Name-Oriented Networking (NOMEN '12), Orlando, FL, USA, March 2012*, pages 280–285.

[18] I. Psaras, W.K. Chai, and G. Pavlou. Probabilistic in-network caching for information-centric networks. In *Proceedings of the 2nd Workshop on Information-Centric Networking, Orlando, FL, USA, March 2012*, pages 55–60.

[19] Anirban Mahanti, Derek Eager, and Carey Williamson. Temporal locality and its impact on web proxy cache performance. *Elsevier Journal on Performance Evaluation*, 42(2):187–203, 2000.

[20] Uichin Lee, Ivica Rimac, and Volker Hilt. Greening the internet with content-centric networking. In *Proceedings of the 1st International Conference on Energy-Efficient Computing and Networking (e-Energy '10), Passau, Germany, April 2010*, pages 179–182.

[21] Tomasz Janaszka, Dariusz Bursztynowski, and Mateusz Dzida. On popularity-based load balancing in content networks. In *Proceedings of the 24th International Teletraffic Congress (ITC'12), Krakow, Poland, September 2012*, pages 1–8.

[22] Jun Li, Hao Wu, Bin Liu, Jianyuan Lu, Yi Wang, Xin Wang, Yanyong Zhang, and Lijun Dong. Popularity-driven coordinated caching in named data networking. In *Proceedings of the 8th ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS '12), Austin, TX, USA, October 2012*, pages 15–26.

[23] Nikolaos Laoutaris, Orestis Telelis, Vassilis Zissimopoulos, and Ioannis Stavrakakis. Distributed selfish replication. *IEEE Transactions on Parallel and Distributed Systems*, 17(12):1401–1413, 2006.

[24] Vasilis Sourlas, Paris Flegkas, Lazaros Gkatzikis, and Leandros Tassiulas. Autonomic cache management in information-centric networks. In *Proceedings of the IEEE Network Operations and Management Symposium (NOMS '12), Maui, HI, USA, April 2012*, pages 121–129.

[25] Alexander Afanasyev, Ilya Moiseenko, and Lixia Zhang. ndnsim: Ndn simulator for ns-3. *Technical Report NDN-0005, Named-Data Networking Project*, October 2012.

[26] Neil Spring, Ratul Mahajan, and David Wetherall. Measuring isp topologies with rocketfuel. *IEEE/ACM Transactions on Networking (TON)*, 32(4):133–145, 2002.

[27] Meeyoung Cha, Haewoon Kwak, Pablo Rodriguez, Yong-Yeol Ahn, and Sue Moon. I tube, you tube, everybody tubes: analyzing the world's largest user generated content video system. In *Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement (IMC '07), San Diego, CA, USA, October 2007*, pages 1–14.

[28] Phillipa Gill, Martin Arlitt, Zongpeng Li, and Anirban Mahanti. Youtube traffic characterization: a view from the edge. In *Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement (IMC '11), San Diego, CA, USA, October 2007*, pages 15–28.

[29] Hongliang Yu, Dongdong Zheng, Ben Y Zhao, and Weimin Zheng. Understanding user behavior in large-scale video-on-demand systems. In *Proceedings of the 1st ACM European Conference on Computer Systems (EUROSYS '06), Leuven, Belgium, April 2006*, volume 40, pages 333–344.

Biography

Andriana Ioannou, is a PhD student in the Distributed Systems Group (DSG), School of Computer Science and Statistics in Trinity College, Dublin. She received her Bachelor degree in Informatics from Aristotle University of Thessaloniki, Greece in 2010. Her current research interests lie on the area of Information-Centric Networking. Email: ioannoa@scss.tcd.ie

Stefan Weber, is a Professor in the School of Computer Science and Statistics in Trinity College, Dublin. He received a Ph.D. from Trinity College, Dublin, in 2002. His current research interests include information-centric networking, emergent networks such as mobile ad hoc networks and protocols and architectures for the next generation of the Internet. Email: sweber@scss.tcd.ie